



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES

#15
LST
11-7-02

Applicant: Brian DONOVAN

Group Art Unit: 2155

Serial N°: 09/410,202

Examiner: David Y. Eng

Filed: September 30, 1999

Title: ZERO OVERHEAD COMPUTER INTERRUPTS WITH TASK
SWITCHING

RECEIVED

OCT 24 2002

Technology Center 2100

APPEAL BRIEF ON BEHALF OF APPLICANT

1600 ODS Tower
601 SW. Second Avenue
Portland, Oregon 97204-3157

October 10, 2002

Commissioner of Patents
Box AF
Washington, D.C. 20231

Dear Sir:

RECEIVED
OCT 30 2002
TC 3700 MAIL ROOM

This is an appeal from the decision of the Examiner in the August 28, 2002 Official Action finally rejecting claims 2 and 4 through 16. The *Notice of Appeal* is filed concurrently herewith and is accompanied by a check for \$320 covering the appeal fee and the filing fee for this brief for a small entity.

REAL PARTY IN INTEREST

The real party in interest is Xyron Corporation, an Oregon corporation.

10/21/2002 BABRAHA1 00000146 09410202

02 FC:2402

160.00 OP

RELATED APPEALS AND INTERFERENCES

There are no known related appeals or interferences.

STATUS OF CLAIMS

Claims 1 and 3 have been cancelled. Claims 2 and 4 through 16 remain in the case and are the claims appealed herein. Claims 2 and 4 through 16 are set forth in the Appendix to this brief.

STATUS OF AMENDMENTS

An amendment under Rule 116 has been filed subsequent to the final rejection cancelling claims 1 and 3 in order to place the case in condition for this appeal. In addition, a second amendment correcting a typographical error in claim 5 has been filed concurrently with the *Notice of Appeal*.

SUMMARY OF THE INVENTION

The present invention is a circuit for managing the execution of tasks in a microprocessor-based computer system which may be monolithic (FIG. 3, generally). In such systems, interrupt signals from various inputs and peripheral devices (90) must be processed in a logical way. Tasks on the CPU bus scheduled to be run by the microprocessor have been initially assigned a priority code by software. Interrupt signals, however, can contain requests for task processing that have higher priorities. At the same time, tasks previously scheduled for orderly execution must eventually be processed and cannot be continuously shunted to the back of the line.

The present invention provides a circuit for the managing of task prioritization

switching and execution in a real time system. The invention thus provides a task enable circuit (81, 91, 83, 97, 96) which determines from predetermined inputs such as interrupts (90) whether a task is ready for execution by a central processing unit. Once a task has been determined as ready for execution, a task priority selection circuit (94, 93, 95, 19, 92) determines the position or order in which the task will be run. Finally, a task switching circuit (20) controls the execution of tasks in the sequence which is determined by the task priority selection circuit. The task priority selection circuit includes a variable rate task priority incrementing circuit (93) that varies the assigned priority of a task so that the priority increases with time. In this way, its execution within a predetermined amount of time is assured regardless of other interrupt signals. Another feature of the invention is a task linking circuit (81) that links together a plurality of tasks in a predetermined order. Thus, in the event of an interrupt which takes precedence over tasks in a linked list, the resumption of the execution of the tasks in the linked list after the processing of the interrupt task will resume in the proper order.

The invention also includes a zero overhead switching circuit (generally FIG. 1) which enables the processing of a task without wasting bus cycles to fetch and load the task (see timing diagram FIG. 2). Tasks are processed for priority and switched into the registers (3 and 4) of the zero overhead circuit (FIG. 1). Multiplexers (13 and 17) appropriately switch the tasks to the CPU (1) or to an SRAM (2). In turn, the SRAM is coupled back to the two registers (3 and 4) by multiplexers (14 and 15). One task is always latched and ready to run, residing in latch (3 or 4) so that no "overhead" is used in fetching and storing that task if it is next in priority for execution.

All these functions take place in logic circuitry that operates between the software and peripheral/hardware interrupts and the microprocessor. Other than the setting of

initial priorities by software, there is no software processing of either the switching or prioritization of tasks presented for execution.

ISSUES

1. Whether claims 2, 4 through 7 and 14 through 16 are unpatentable over Madnick under 35 U.S.C. §103.
2. Whether claim 8 is unpatentable under 35 U.S.C. §112, second paragraph.
3. Whether claim 8 is unpatentable over Madnick in view of George under 35 U.S.C. §103(a).
4. Whether claims 9 and 10 through 13 are unpatentable over Madnick in view of Jen under 35 U.S.C. §103(a).

GROUPING OF CLAIMS

Claims 2 and 14 stand or fall together.

Claims 4 through 9 and 15, 16 stand or fall together.

Claims 10 through 13 stand or fall together.

ARGUMENT

The principal reference used by the Examiner in rejecting all of the claims of the instant application is a portion of a textbook written by Madnick. Specifically, the Examiner relies on Chapter 4 of this textbook entitled *Processor Management*. The book was written in 1974, well before the advent of the microprocessor. The chapter on processor management relates to procedures for the orderly organization of jobs to be run by the data processing core in a way that made the most efficient use of system resources.

Typically, such jobs were in the form of machine language code loaded into the data processing machine by way of stacks of punch-cards or rolls of reel-to-reel tape. Jobs are processed in this way in batches, that is, the jobs are arranged in an orderly queue. The chapter on processor management is an attempt to create various rules for the processing of these batch jobs that depend on factors such as the amount of time required, the relative priority of the job, and the memory constraints placed on the system by the job. In all these cases however, the rules are applied to human decision making as to which jobs will be allowed to be processed by the core system.

Several basic algorithms for choosing which jobs to run are summarized in section 4-2.5. The author notes that there are primarily three basic algorithms: first in, first out; shortest job first; and future knowledge. Regardless of methodology however, all of the procedures discussed in Madnick's textbook relate to a selection of a batch of programming instructions which will be allowed to process one at a time.

By contrast, all of the claims of the instant application relate to a circuit which automatically, and according to its own internal logic, selects tasks (not jobs) for processing. Tasks are individual statements in machine code that are executed by a central processing unit. Thus, the claims call for a circuit. Claim 2, for example, contains as element (a)

"providing an integrated circuit having circuit components for automating the selection of tasks to be performed by said computer system;"

Claims 4 through 13 and 15, 16 are apparatus claims which specifically claim an interrupt and task change processing circuit.

Madnick does not disclose an electronic circuit for performing any of the management and processing functions that it describes. By contrast, all of these functions

are done by humans according to a set of rules or decision making protocol. The other difference is that in the Madnick textbook, real time tasks are not being dealt with by this protocol. Madnick shows different rules for batch processing only. This is completely different from the circuit of the claimed invention which operates in real time with elements like computer peripheral devices issuing interrupt commands while the processor is processing data. There are no peripheral devices in the computers Madnick describes.

Thus, other than a general recognition that a computer with limited processing resources must have some kind of job allocation procedure, there is nothing in Madnick that is in any way similar to the subject matter of the instant claims.

The Examiner attempts to cover the deficiency in the Madnick reference by referring generally to patents to Jen and George. George, however, is a disk control system. In George, various processing jobs are stored on a disk and are moved from the disk to the central processing unit according to rules that are specified in software. Reference to FIG. 1 of George is especially instructive. There is a CPU in main memory but all of the other hardware resides in the form of disk controllers and disk drives. The flowcharts in FIGs. 6 through 20B are software routines. The same is true of Jen (U.S. patent N° 3,789,365) cited by the Examiner in combination with Madnick in rejecting claim 9. Jen describes a Univac computer which uses software to move interrupted tasks in and out of memory units that are associated with processor functions.

There is no suggestion in either Madnick, George or Jen that suggests the combination that the Examiner wishes to make. George and Jen are data processing systems. Madnick is a textbook that deals with human procedures for making efficient use of data processing systems. None of these references are combinable with each other because, first, nowhere does Madnick suggest that its procedures could be implemented

automatically in a hardware circuit that responds to real time interrupts and nowhere in George or Jen is there any appreciation for the fact that their own functions might be capable of hardware implementation without the software control that is necessary to stop, start and replace processing functions while meanwhile moving large blocks of data and/or coded instruction to off-processor memory locations.

Claim 4 recites in its preamble that it is for a microprocessor-based computing system having a CPU for executing tasks. This element cannot be found in any of the prior art of record. The claim further requires for task register sets and includes peripheral devices that issue interrupt commands. Neither Madnick nor George has this feature. The elements of claim 4 include a task enable circuit, a task priority selection circuit, and a task switching circuit. None of these elements can be found in Madnick, George, Jen or any combination thereof. There is nothing in any of the prior art, for example, that comprises a circuit that interprets interrupt signals, looks at the task presented by the interrupt signal, and determines whether or not the task is ready for execution by the central processing unit. Further, in none of the prior art is there a priority selection circuit coupled to an output of the task enable circuit which determines an order for the running of tasks determined ready for execution. Finally, there is no task switching circuit in any of the prior art that controls the execution of tasks in a sequence determined by the task priority selection circuit.

Claim 10 includes the same preamble as claim 4 and includes an interrupt and task change processing circuit which responds to interrupt commands and places tasks in an order of priority for execution. Claim 10 also includes a zero overhead multiplexing circuit. This circuit stores a later task in a set of latches during a first clock cycle and simultaneously switches a previously stored task stored in a second set of latches into a

memory unit during the same clock cycle. The Examiner has not even attempted to show that there exists anything like this circuit. It is clearly not taught by Madnick (which has no circuits at all) or George or Jen (which rely on software for the control of tasks presented to the data processing unit).

In the Office Action of August 28, 2002, the Examiner again rejected the same claims based upon the same prior art as discussed above. In addition however, the Examiner cited two new patents – Magar (U.S. patent N° 4,507,727) and Rueben (U.S. patent N° 4,628,158). These references were cited apparently because they disclose “logic circuits.” Without saying why, the Examiner makes the leap of logic (for which there is no support in the references themselves) that the mere existence of logic elements in these patents somehow supplies the Madnick reference with the needed circuitry to implement task scheduling and switching functions. In the last sentence, the Examiner makes the astounding statement that

“It can be seen that Macnicks [sic] task schedulaer [sic] is also implemented by IC chips which are made up of logic components which are made up of transistors or gates.”

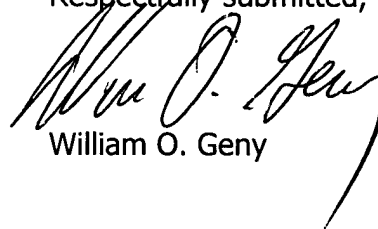
But there is no such structure in Madnick. Whatever job scheduling is done for the data processing system described in Madnick is done by humans making decisions about which stacks of punch cards are to be loaded into the processing queue and when. The mere presence of patents similar to Rueben and Magar does not change the fact that Madnick describes a hand-operated system. Unless some recognition could be found in either of these two prior art references that procedures described in Madnick could be implemented in specific hardware, the Examiner’s assertion must fail.

The Examiner also rejected claim 8 under 35 U.S.C. §112, but the basis for this rejection is not understood. Claim 8 adds an element to claim 4 – a trace enable circuit.

This provides the circuit of claim 4 with additional functionality. Section 112 does not require that a distinct functionality or connection to the rest of the circuit be expressly set forth.

The Examiner has failed to bring forth either prior art or evidence which teaches the invention covered by the claims at issue. For the reasons stated above therefore, the claims should be allowed and the case should pass to issue.

Respectfully submitted,

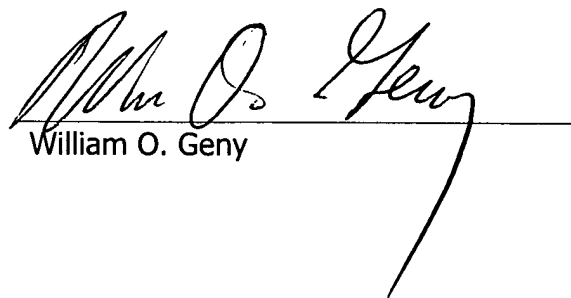


William O. Geny

CERTIFICATE OF MAILING

I hereby certify that this *APPEAL BRIEF ON BEHALF OF APPLICANT* is being deposited with the United States Postal Service as first class mail on October 10, 2002 in an envelope addressed to: Commissioner for Patents, Box AF, Washington, D.C. 20231.

Dated: October 10, 2002



William O. Geny



**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES**

Applicant: Brian DONOVAN

Group Art Unit: 2155

Serial N^o: 09/410,202

Examiner: David Y. Eng

Filed: September 30, 1999

Title: ZERO OVERHEAD COMPUTER INTERRUPTS WITH TASK
SWITCHING

APPENDIX TO APPEAL BRIEF ON BEHALF OF APPLICANT

Claims

RECEIVED
OCT 24 2002
Technology Center 2100

1. Cancelled by amendment of October 2, 2002.

2. A method for ordering the performance of tasks in a computer system, said computer system having input sources that issue interrupt signals for requesting the performance of a task, said method comprising:

- (a) providing an integrated circuit having circuit components for automating the selection of tasks to be performed by said computer system;
- (b) wherein the integrated circuit performs the following steps:
 - (i) assigning a priority level for each task based upon a selected parameter of an interrupt signal;
 - (ii) changing each priority level as a function of time; and
 - (iii) beginning the execution of a first task when said priority level of said first task exceeds said priority level of all other tasks.

3. Cancelled by amendment of October 2, 2002.

RECEIVED
OCT 30 2002
TC 3700 MAIL ROOM

interrupt commands, an interrupt and task change processing circuit comprising:

- (a) a task enable circuit for determining from predetermined inputs whether a predetermined task is ready for execution by the central processing unit,
- (b) a task priority selection circuit coupled to an output of the task enable circuit for determining an order for the running of tasks that have been determined ready for execution by the task enable circuit; and
- (c) a task switching circuit coupled to an output of the task priority selection circuit for controlling the execution of tasks in a sequence determined by the task priority selection circuit.

5. The interrupt and task change processing circuit of claim 4 wherein said task priority selection circuit includes a variable rate task priority incrementing circuit for varying an assigned priority of a task such that said priority increases with time, whereby its execution is caused to occur within a predetermined period of time.

6. The interrupt and task change processing circuit of claim 4 wherein the task enable circuit includes a task linking circuit for linking together groups of tasks which are dependent upon each other.

7. The interrupt and task change processing circuit of claim 4 wherein the task enable circuit is responsive to a task interrupt signal for designating a task as ready to run, said task enable circuit including a timer for generating said task interrupt signal after a predetermined period of time.

8. The interrupt and task change processing circuit of claim 4 further including a trace enable circuit for recording register states of selected registers during a preselected clock cycle.

9. The interrupt and task change processing circuit of claim 4 wherein the task switching circuit is coupled to a zero overhead multiplexing circuit for storing a later task in a first set of latches during a first clock cycle while simultaneously switching a previously stored earlier task stored in a second set of latches into a task switch controller during the

9. The interrupt and task change processing circuit of claim 4 wherein the task switching circuit is coupled to a zero overhead multiplexing circuit for storing a later task in a first set of latches during a first clock cycle while simultaneously switching a previously stored earlier task stored in a second set of latches into a task switch controller during the same clock cycle.

10. In a microprocessor-based computing system having a CPU for executing tasks represented by task register sets and further including peripheral devices that issue interrupt commands, the combination comprising:

- (a) an interrupt and task change processing circuit for responding to interrupt commands and for placing tasks in an order of priority for execution by the CPU, and
- (b) a zero overhead multiplexing circuit coupled to the interrupt and task change processing circuit for storing a later task in a first set of latches during a first clock cycle while simultaneously switching a previously stored earlier task stored in a second set of latches into a memory unit during the same clock cycle.

11. The combination of claim 10 wherein the interrupt and task change processing circuit includes a task enable circuit for placing a task in a status in which it is ready for execution by the CPU.

12. The combination of claim 11 wherein the interrupt and task change processing circuit includes a task priority selection circuit for assigning a task priority to tasks which are ready for execution by the CPU.

13. The combination of claim 12 wherein the interrupt and task change processing circuit includes a task switching circuit for loading tasks ready for execution by the CPU into said zero overhead multiplexing circuit based upon their task priority.

14. The method of claim 2 wherein step (b)(ii) is accomplished at different rates

of time for different tasks.

15. The circuit of claim 4 wherein said task priority selection circuit includes means for assigning a priority level for each task and timing means for changing said priority levels as a function of time.

16. The circuit of claim 15 wherein said timing means includes means for varying the rate of change of said priority levels as a function of time.